PROCEDURES AND TEMPLATES FOR TEST CREATION

Release Date: November 8, 2000

Contract Number: GS-35F-4919H
Order Number: T0000AJ3739

CLIN #009-1

Prepared for:

DEPARTMENT OF EDUCATION
OFFICE OF STUDENT FINANCIAL ASSISTANCE (SFA)
400 Maryland Avenue, SW
Washington, DC 20202

Prepared by:

CTGi
Suite 250
10461 White Granite Drive
Oakton, VA 22124

This page intentionally left blank

FOREWORD

CTG, Incorporated (CTGi), would like to thank the following personnel whose dedication and involvement made the development and completion of this document possible.

Kelly Conboy

Glenn Foster

Candace Jones

Frank Majewski

Annette Mazie

Antony Mosley

Isaac Sanvee

Paul Stocks

Sandra Stocks
Director Of Operations,
Washington Metropolitan Area
CTGi

This page intentionally left blank

EXECUTIVE SUMMARY

This document has been prepared for the Department of Education, Office of Student Financial Assistance (SFA) by CTG, Incorporated (CTGi), for the purpose of developing a standardized procedure for the creation of test conditions, test scenarios, and test data.

After final approval, by SFA, this document, along with those listed below, will be integrated into and become the final deliverable of this contract, Department of Education, SFA, System Integration and Testing (SI&T) Process Handbook. The Department of Education, SFA, SI&T Process Handbook will then become integrated into the overall Department of Education, SFA Modernization Technology Handbook. The remaining documents that will comprise the SI&T Process Handbook are:

- System Integration and Testing Standards
- Test Performance Measurements
- Procedures and Templates for Test Execution, Test Evaluation, and Error Correction
- Procedures and Templates for System Configuration Management (CM) and Quality Control (QC)
- Procedures For Using Testing Tools

The above listed procedures, templates, and guide are being prepared for delivery as a separate documents.

All SI&T guidelines and procedures are focused on the support, development, and execution of a comprehensive SI&T processes. To this end, the document contains information on understanding issues related to test planning roles, responsibilities of the organization, and required testing practices. The standards and procedures used in the support of this document are reflective of industry best practices, practices of other federal government agencies, and various governing standards and literature regarding the integration and testing process.

This page intentionally left blank

TABLE OF CONTENTS

This page intentionally left blank

LIST OF ACRONYMS

CI	Configuration Item
CM	Configuration Management
CSCI	Computer Software Configuration Item
CTGi	CTG, Incorporated
FEDSIM	Federal Systems Integration and Management Center
HWCI	Hardware Configuration Item
IEEE	Institute of Electrical and Electronics Engineers
QA	Quality Assurance
QC	Quality Control
SFA	Office of Student Financial Assistance
SI&T	System Integration and Testing
SQT	System Qualification Test
SRD	Software Requirements Document
STD	System Test Description
STP	System Test Plan

This page intentionally left blank

1. INTRODUCTION

1.1 Background

The United States Department of Education, Office of Student Financial Assistance (SFA), contracted CTG, Incorporated (CTGi) in August 2000 to develop standardized System Integration and Test (SI&T) procedures that will be used for guidance, planning, and implementation of current and future Department of Education enterprise information technology systems projects.

1.2 Objective

The document's objective is to provide procedures and templates for standardization of test conditions, test scenarios, and compilation of necessary test data. The information contained in the procedures and templates is necessary to prepare for the execution of all types of Department of Education, SFA, information technology system testing.

The creation of standardized procedures for creating testing conditions is critical to establishing a consistent and comprehensive test capability. These standard procedures will be documented and available to all personnel involved in the oversight and testing of software applications. The purpose of these procedures is to promote a consistent approach in the development and testing of application products. The standard integration and testing processes include the essentials of what must be done, why, and who performs and completes the work. Procedural activities are provided to explain how to complete the process (the sequence of tasks or task steps that have to be performed), when the work is performed and completed, and the criteria for measuring the quality of the work.

1.3 Applicability

When the SI&T process is performed by either the U. S. Department of Education, SFA, staff and/or contractors this document will apply, unless specifically excluded, in the program/project plan, contract, etc. This document will be used for the creation of guidelines and procedures for the planning, preparation, execution, analysis, and evaluation, of all types of U. S. Department of Education, SFA, information technology project integration and testing.

1.4 Document Organization

This document contains two narrative sections, a Glossary, a Bibliography, and two appendices. Section 1, Introduction, provides brief background information and states the guiding objective and applicability for the document. Section 2, Procedures and Templates For Testing Creation, provides the basis and procedures for the creation of a System Test Plan (STP) template and a System Test Description (STD) template needed for proper testing procedures and test conditions. Appendix A provides the template for creating a STP. Appendix B provides the template for creating a STD.

2. PROCEDURES FOR CREATION OF TESTING CONDITIONS

Establishing standardized procedures for application development processes is critical to the establishing a consistent and comprehensive test capability. These standard procedures must be documented and made available to all personnel involved in the integration and testing process. The purpose of these standard procedures is to promote a consistent approach in both the development and testing of application products. The standard testing processes include essentials on what has to be done, why, and who performs and completes the work. Procedural activities are provided to explain how to complete the process (e.g., the sequence of tasks or task steps to be performed), when the work is performed and completed, and the criteria for measuring the quality of the work.

The following items are considered paramount in the process of defining standard procedures:

- Provide, in as much detail as possible, a description of how the test engineers will develop and generate test artifacts. Describe test case development as a series of steps that application engineers can follow.

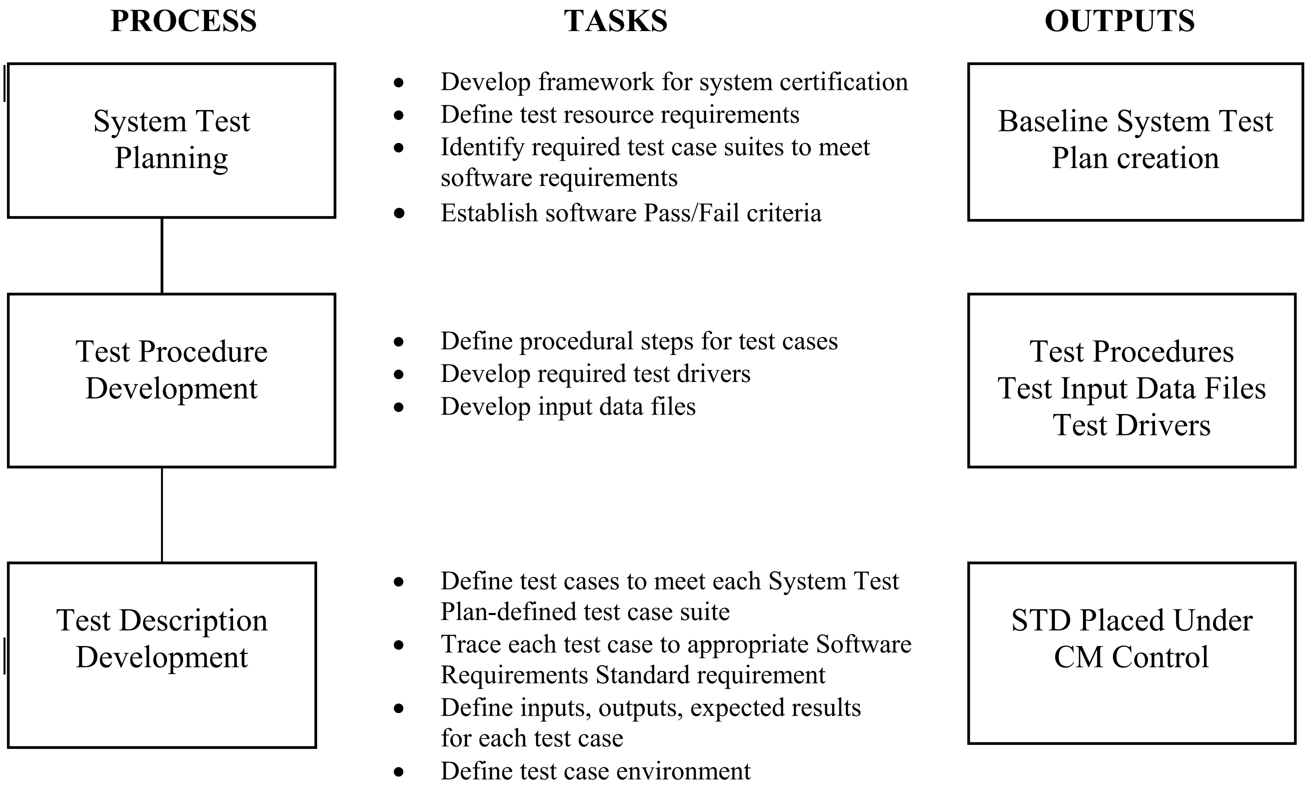
- Ensure test case procedures are a series of repeatable sets of steps. Make each step of a test case procedure as mechanical as possible. This helps to eliminate ambiguity and determine which portions can be automated.

- Provide an overview that helps test engineers understand the types of work products in the test environment. The components of the STP are used for this overview.

- Identify standards for form and content of test engineering work products. Standardization of form and content is critical.

2.1 Determining Software Test Execution Conditions

The activities associated with developing a comprehensive testing approach are presented in Figure 2-1.



PROCESS	TASKS	OUTPUTS
System Test Planning	• Develop framework for system certification • Define test resource requirements • Identify required test case suites to meet software requirements • Establish software Pass/Fail criteria	Baseline System Test Plan creation
Test Procedure Development	• Define procedural steps for test cases • Develop required test drivers • Develop input data files	Test Procedures Test Input Data Files Test Drivers
Test Description Development	• Define test cases to meet each System Test Plan-defined test case suite • Trace each test case to appropriate Software Requirements Standard requirement • Define inputs, outputs, expected results for each test case • Define test case environment	STD Placed Under CM Control

Figure 2-1. Procedures For Creation Of System/Software Test Conditions

2.2 System Test Plan Development

The STP describes procedures for qualification testing of Computer Software Configuration Items (CSCI) and software systems. It describes the software test environment to be used for the testing, identifies the tests to be performed, and provides schedules for test activities. There is usually a single STP for a system or project. The STP enables the customer to assess the adequacy of the planning for CSCI and all other test phases of the SI&T process.

Irrespective of the origin of the application(s) software, the software requirements drive the test activities. A testing strategy should include automated mechanisms to support the effective and repeatable performance of test-related activities. The processes that support the testing are documented to create a repeatable process that will ensure quality in the developed work products.

The STP employs the activities for Integration Test (i.e., both Regression and CSCI), Performance Test, and System Qualification Test (SQT) (i.e., user acceptance testing) support to develop a strategy that will produce the highest quality product to meet the customer's system requirements.

2.2.1 Participants

Participants will consist of the government personnel designated/assigned by SFA and/or the SI&T Project Manager. The participants will be responsible for overseeing the activities to be performed, and providing documentation and answers to the SI&T team (contractor and/or government). The participants should have knowledge and/or understanding of the system to be tested.

Participants may include the Project Manager of the system being tested, the test manager and/or engineer, a Quality Assurance (QA) group or person, a system development group or person, and a CM group or person. The size and composition of the group of participants will vary from system to system due to system complexity, size, function, etc.

2.2.2 Entrance Criteria

Approved software requirements contained in the baseline Software Requirements Document (SRD)

2.2.3 Inputs

Baseline SRD

2.2.4 Activities

a. Review software performance requirements documented in the SRD. It is recommended that the requirements be organized into a hierarchical structure based on system functions.

b. Define test case suites, test classes (e.g., path analysis, stress, capacity, and timing), and test methodologies (e.g., static, dynamic). Following the hierarchical structure of the software requirements can facilitate the determination of test case suites.

c. Identify the environment in which testing will be conducted.

d. Define plans for implementing and controlling the test environment.

e. Estimate the personnel and other resources required for implementing the test concept and objectives.

f. Assign a project test leader and support personnel to prepare, conduct, analyze, and report the results of testing.

g. Develop a schedule of defined tests case suites for inclusion in the STP. Including time for problem correction and retest

h. Create the STP, following the instructions of the assigned documentation standard.

i. Conduct a peer review.

j. Revise draft STP to correct discrepancies and incorporate recommended changes.

k. Publish final STP. The process includes approval, distribution, and placing the STP under baseline control in the Configuration Management (CM) Software Library.

2.2.5 Outputs

Baseline STP

2.2.6 Exit Criteria

Baseline STP placed under CM control.

2.3 System Test Description Development

The baseline STD describes test preparations, test cases, and test case procedures used to perform qualification testing of a CSCI or a software system or subsystem. The baseline STD enables the customer to assess the adequacy of the qualification testing to be performed.

2.3.1 Participants

Participants will include government personnel designated/assigned by the SFA and/or the SI&T Project Manager. The participants will be responsible for overseeing the activities to be performed and providing documentation and answers to the SI&T team (contractor and/or government). Participants should have knowledge and/or understanding of the system to be tested.

Titles of the participants may include the Project Manager of the system being tested, a test manager and/or engineer, a QA group or person, a system development group or person, and a CM group or person. The size and composition of the participants will vary from system to system due to system complexity, system size, system function, etc.

2.3.2 Entrance Criteria

Test planning has resulted in a STP.

2.3.3 Inputs

a. STP.

b. Baseline SRD.

2.3.4 Activities

a. For each test case suite, identify the required individual test cases. Following the hierarchical structure of the software requirements for each functional area of the system can facilitate identification of the test cases.

b. For each test case, define the inputs (stimuli) required to fulfill the test purpose. See Section 2.4 of this document for details.

c. For each test case, define the expected results (i.e., output from the test in response to input) that serve as the test case Pass/Fail criteria.

d. Define insertion and extraction methods to/from test. Identify points of data input/output and volumes of data.

e. Define evaluation criteria for test results analysis. Define ranges of values, capacities, and times for test Pass/Fail status.

f. Trace SRD performance requirements to specific test cases. Verify the completeness of performance requirement(s) allocation.

g. Identify test environment configuration, interface drivers, database loaders, controllers/monitors, and other test tools to support test case purposes.

h. Draft test case suite descriptions in a STD document.

i. Conduct a peer review.

j. Revise draft to correct discrepancies and incorporate recommended changes.

2.3.5 Outputs

A STD that implements the test strategy and includes defined test cases and test case procedures.

2.3.6 Exit Criteria

a. Baseline STD placed under CM control.

b. All SRD requirements are traceable to test cases.

2.4 Procedures For Creating Test Case Procedures

Activities include developing detailed steps for each test case. The results of these test case procedures are incorporated into Section 4.2.1.6 of the baseline STD (see Appendix B).

2.4.1 Participants

Participants will consist include government personnel designated/assigned by the SFA and/or the SI&T Project Manager. The participants will be responsible for overseeing the activities to be performed and for providing documentation and answers to the SI&T team (contractor and/or government). Participants should have knowledge and/or understanding of the system to be tested.

Titles of the participants may include the Project Manager of the system being tested, the test manager and/or engineer, a QA group or person, a system development group or person, and a CM group or person. The size and composition of the group of participants will vary from system to system due to system complexity, size, function, etc.

2.4.2 Entrance Criteria

Test cases identified in a STD.

2.4.3 Inputs

a. STD

b. Requirements allocated to individual test cases.

c. Available draft user and operator manuals or checklists.

2.4.4 Activities

a. Review software user and operator manuals to identify methods of operator input, use of simulator/emulator tools, and software data recording.

b. Define detailed test steps for providing inputs for test cases.

c. Define measurable detailed test results for test steps.

d. Prepare input data file(s) to provide test stimuli.

e. Define evaluation steps for conducting post-test analysis and comparison of actual and expected test results.

f. Draft test case procedures.
g. Conduct a peer review.
h. Revise draft test case procedures to correct discrepancies and incorporate recommended changes.

2.4.5 Outputs

a. Documented test case procedures, including inputs and expected results.
b. Test input data file(s) compiled.

2.4.6 Exit Criteria

Approved test case procedures for all test cases in the baseline STD.

This page intentionally left blank

GLOSSARY

Aggregate

A mass of distinct things gathered into a total or whole.

Aggregation Level

Effective measurement analysis and reporting requires that the data be aggregated to higher levels of the of the software components and project organizational structure. The aggregation levels define the different ways the measurement data can be grouped and organized for reporting on the project. The aggregation levels describe how the measurement data relates to an existing product and process structures. The organization that allows the measurement results to be combined, and later decomposed, into meaningful pieces of information.

Aggregation Structure

The structure used to define the data according to the defined aggregation levels. The levels may describe the personnel and management structure of the project, or the configuration of physical components of the project. All entries in a structure should be of the same type, such as software modules. However, these entries may reside at various levels of the structure, such as software modules at the unit level, CSCI, or integrated level of the software architecture.

Application

(1.) A complete, self-contained program that performs specific function(s) directly for the user.

(2.) In the TPM process this term refers to one of the two basic measurement activities which comprise the system measurement process. The application activity involves collecting, analyzing, and acting upon the measurement data.
See **Tailoring**.

Automated Test Script

A computer readable set of instructions that performs a sequence of steps, sub-steps, or other actions, performed serially, in parallel, or in some combination of consecution, that creates the desired test conditions that the test case is deigned to evaluate.

Baseline

A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures.

Baseline Control

Baseline control is the process that regulates approved and released versions of all software, documentation, and the test environment throughout the test life cycle.

Black Box Testing

This is testing associated with functional testing where the object being tested is treated as a black box. In this type of testing the test object is subjected to inputs and outputs that are verified for conformance to prescribed specifications.

Capacity Testing

Attempts to simulate expected customer peak load operations in order to ensure that the system performance requirements are met. It does not necessarily exercise all of the functional areas of the system, but selects a subset that is easy to replicate in volume. It will ensure that functions which are expected to use the most system resources are adequately represented.

Change Control

The process by which problems and changes to the software, documentation, and test environment are evaluated, approved, rejected, scheduled, and tracked.

Computer Aided Software Engineering (CASE)

A technique for using computers to help with one or more phases of the software life cycle, including the systematic analysis, design, implementation and maintenance of software. Adopting the CASE approach to building and maintaining systems involves software tools and training for the developers who will use them.

Computer Software Configuration Item (CSCI)

An aggregation of software that is designated for configuration management and treated as a single entity in the configuration management process.

Configuration Control

An element of configuration management, consisting of the evaluation, coordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification.

Configuration Item (CI)

Hardware or software, or an aggregate of both, which is designated by the project configuration manager (or contracting agency) for configuration management.

Configuration Management (CM)

A discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements.

Configuration Management Office (CMO)

The Configuration Management Office (CMO) is the officiator of the project life cycle CM process.

Criteria

A standard, rules, or tests by which something can be judged.

Critical Defect

See Criticality

Criticality

The assessment of the impact upon a system of a given error, defect, problem, or discrepancy during the life cycle of a system.

The definition of critical and non-critical system defects or problems should be addressed at a management level and can be different for each system. For any given system error, defect, problem, or discrepancy, an appropriate impact value (i.e., priority) will be assigned.

An example of impact values with the corresponding priority numbers is presented below as contained in IEEE/EIA Std-12207, 1998. The priority that will apply if a problem can result in one or more of these impacts:

PRIORITY IMPACT

1. a.) Prevent the accomplishment of an operational or mission essential capability.

   b.) Jeopardize safety.

   c.) Cause significant technical, cost, or schedule risks to the project or to life cycle support of the system.

2. a.) Adversely affect the accomplishment of an operational or mission essential capability and no work-around solution is known.

   b.) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, and no work-around is known.

3. a.) Adversely affect the accomplishment of an operational or mission essential capability, but a work-around solution is known.

   b.) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, but a work-around is known.

4. a.) Results in user/operator inconvenience or annoyance, but does not affect a required operational or mission essential capability.

   b.) Results in inconvenience or annoyance for development or support personnel, but does not prevent the accomplishment of the responsibilities of these personnel.

5. a.) This priority denotes any other effect.

Customer

The organization that procures software systems for itself or another organization.

Developer

An organization that develops software products. The term “develop” may include develop, modification, integration, reengineering, sustaining engineering, maintenance, or any other activity that results in software products. The developer may be a contractor or a government agency.

Discrepancy

An inconsistency or disagreement found during testing between the actual and expected test results.

Document

A data medium and the data recorded on it that generally has permanence and can be read by a human operator or machine. Often used to describe human readable items only (e.g., technical documents, design documents, requirements documents, etc.).

Documentation

(1.) A collection of documents on a given subject.

(2.) The management of documents, that includes the actions of identifying, acquiring, processing, storing, and disseminating.

(3.) Any written or pictorial information describing, defining, specifying, reporting or certifying activities, requirements, procedures, or results.

Driver

A software program that exercises a system or system component by simulating the activity of a higher level component.

Emulation

One system is said to emulate another when it performs in exactly the same way, though perhaps not at the same speed. A typical example would be the emulation of one computer by (a program running on) another. You might use emulation, as a replacement for a system whereas you would use a simulation if you just wanted to analyze it and make predications about it.

Emulator

Hardware or software that performs emulation.

Entry Criteria

A set of decision making guidelines used to determine whether a system under test is ready to move into, or enter, a particular phase of testing. Entry criteria tend to become more rigorous as the test phases progress.

Environment

The infrastructure in which a system is executing, consisting of hardware, operating system software, interfaces, etc.

Exit criteria

A set of decision-making guidelines used to determine whether a system under test is ready to exit a particular phase of testing. When exit criteria are met, either the system under test moves on to the next test phase or the test project is considered complete. Exit criteria tend to become more rigorous as the test phases progress.

Final System Test Report (FSTR)

Used to determine whether system testing is completed and to assure that software is ready for production.

Hardware Configuration Item (HWCI)

An aggregation of hardware that is designated for configuration management and treated as a single entity in the configuration management process.

Independent Verification and Validation (IV&V)

The verification and validation of a software product by an organization that is both technically and managerially separate from the organization responsible for developing the product.

Indicator

A measure or combination of measures that provides insight into a system issue or concept. TPM frequently uses indicators that are comparisons, such as planned versus actual measures. Indicators are generally presented as graphs or tables.

Integration

Combining software or hardware components or both into an overall system.

Integration Testing

The period of time in the software lifecycle during which the application is tested in a simulated production environment to validate the communications and technical architecture of the system. This test phase occurs when all the constituent components of the system under test are being integrated.

Interactive Development Environment (IDE)

A system for supporting the process of writing software. Such a system may include a syntax-directed editor, graphical tools for program entry, and integrated support for compiling and running the program and relating compilation errors back to the source code.

Interface

(1.) A shared boundary (e.g., a hardware component linking two devices or registers, or a portion of storage accessed and/or modified by two or more computer programs).

(2.) To interact or communicate with another system component.

Interface Requirement

A requirement that specifies a hardware, software, or database element with which a system or system component must interface, or that sets forth constraints caused by such an interface.

Interface Specification

A specification that sets forth the interface requirements for a system or system component (e.g., the software interface specification document).

Interface Testing

Tests conducted to ensure that program or system components correctly pass data and/or control to one another.

Issue

An area of concern where obstacles to achieving program objectives might arise. Issues include risks, problems, and lack of information. These three types of issues are defined as:

- Risk -- An area of concern that could occur, but is not certain. A risk is a potential problem. Risks represent the potential for the realization of unwanted, negative consequences from a project event. For example, a project plan may be based on the assumption that a COTS component will be available on a given date. There is a possibility (probability) that the COTS may be delayed and have some amount of negative impact on the project.
- Problem -- An area of concern that a project is currently experiencing or is relatively certain to experience. For example, a shortage of staff with the right skills may be an actual problem that is delaying the project.
- Lack of Information -- An area where the available information is inadequate to reliably predict project impact. Thus, satisfaction of project objectives is questionable even if no problems or risks are present. For example, lack of information about the size of the software to be developed could result in the project "discovering" that it has more work to do than originally planned.

Measure

The result of counting or otherwise quantifying characteristics of a process or product. Measures are numerical values assigned to system attributes according to defined criteria.

Measured (or actual) Value

Actual, current measurement data, such as hours of effort expended or line of code produced.

Measurement

The process of assigning quantitative values of system properties, according to some defined criteria. This process can be based on estimation or direct measurement. Estimation defines planned or expected measures. Direct measurement results in actual measures.

Measurement Analysis

The uses of measurement data to identify problems, assess problem impact, project an outcome, or evaluate alternatives related to system issues.

Measurement Analyst

The person(s) or team responsible for tailoring and applying system measures for a given project or task.

Measurement Information

Knowledge derived from analysis of measurement data and measurement indicators.

Milestone

A scheduled event for which some project or task member or manager is held accountable. A milestone is often used to measure progress.

Module

A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading.

Note: *The terms 'module', 'component', and 'unit' are often used interchangeably or defined to be sub-elements of one another in different ways depending on the context.*

Non-Critical Defect

See Criticality

Performance Testing

The period of time in the system or software development lifecycle during which the response times for the application are validated to be acceptable. The tests ensure that the system environment will support production volumes, both batch and on-line.

Priority

A measure of the elements of importance related to the repair of a system problem that are not considered in defining the severity of a system problem.

Project Manager (PM)

The official responsible for acquiring, developing, or supporting a system to meet technical, cost, schedule, and quality requirements. Acquisition, development, and support will include both internal tasks and work that is contracted to another source.

Quality Assurance (QA)

A planned and systematic pattern of all actions necessary to provide adequate confidence that the product optimally fulfils customers expectations.

Quality Control (QC)

The assessment of product compliance. Independently finding deficiencies assures compliance of the product with stated requirements.

Requirement

(1.) A condition or capability needed to solve a problem or achieve an objective.

(2.) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document. The set of all requirements forms the basis of development.

Regression testing

Part of the test phase of software development where, as new modules are integrated into the system and the added functionality is tested, previously tested functionality is re-tested to assure that no new module has corrupted the system.

Risk

An area of concern that may occur, but is not certain. A risk is a potential problem. Risks represent the potential for the realization of unwanted, negative consequences from a project event. For example, a project plan may be based on the assumption that a commercial off the shelf (COTS) component will be available on a given date. There is a possibility (probability) that the COTS may be delayed and have some amount of negative impact on the project.

Severity

The degree to which a problem adversely influences the system's operation or the overall test effort.

Simulation

Attempting to predict aspects of the behavior of a system by creating an approximate (mathematical) model of it. This can be done by physical modeling, by writing a special-purpose computer program or using a more general simulation package, aimed at a particular kind of simulation. Typical examples are aircraft simulators or electronic circuit simulators.

Simulator

Hardware or software that performs simulation.

Software Design Specification (SDS)

A document that records the design of a system or system component; typical contents include: system and/or component algorithms, control logic, data structures, data set use, input/output formats, and interface descriptions.

Software Development File (SDF)

The developer shall document the development of each Computer Software Unit (CSU), Computer Software Component (CSC), and CSCI in Software Development Files (SDF). The developer shall establish a separate SDF for each CSU or a logically related group of CSUs, for each CSC or a logically related group of CSCs, and for each CSCI. The developer shall document and implement procedures to establish and maintain SDFs. SDFs may be generated, maintained, and controlled by automated means. To reduce duplication, SDFs should not contain information provided in other documents or SDFs. The set of SDFs shall include (directly or by reference) the following information:

- Design considerations and constraints.
- Design documentation and data.
- Scheduling and status information.
- Test requirements and responsibilities.
- Test case, test case procedures, and results.

Software Life Cycle

The phases a software product goes through between when it is conceived and when it is no longer available for use. The software life cycle typically includes the following: requirements, analysis, design, construction, testing (validation), installation, operation, maintenance, and retirement. The development process tends to run iteratively through these phases rather than linearly; several models (spirals, waterfall, etc.) have been proposed to describe this process. Other processes associated with a software product are: quality assurance, marketing, sales, and support.

Software Management Plan

A project plan for the development of the software component of a system or for the development of a software product.

Software Requirements Document (SRD)

This is a formal document derived from the Software Requirements Specification (SRS) that sets forth the requirements, specifications, and standards for a system (e.g., a software product). Typically included are functional specifications and requirements, performance specifications and requirements, interface specifications and requirements, design specifications and requirements, and development requirements and standards.

Software Requirements Specification (SRS)

A specification that sets forth the requirements for a system component; (e.g., a software product). Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards.

Software Tool

Computer programs used to help develop, test, analyze, or maintain another computer program or its documentation.

Specification

Documentation containing a precise, detailed, verifiable description of particulars with respect to the requirements, design, function, behavior, construction, or other characteristics of a system or system component.

Stub

(1.) A dummy procedure used when linking a program with a run-time library. The stub routine need not contain any code and is only present to prevent "undefined label" errors at link time.

(2.) A local procedure in a remote procedure call (RPC). The client calls the stub to perform some task and need not necessarily be aware that RPC is involved. The stub transmits parameters over the network to the server and returns the results to the client/caller.

System

(1.) Any large program.

(2.) The entire computer system, including the input/output devices, supervisor program or operating system and possibly other software.

System Problem Report (SPR)

A form that is used to record a discrepancy discovered during the Integration Test, Performance Test and System Qualification Test phases of the SI&T process concerning a Computer Software Configuration Item, a software system or subsystem, other software related items, and associated documentation.

System Problem Report (SPR) Status Report

The System Problem Report Status Report is used during the SPR Status Review to determine if the SPRs are being processed appropriately and expeditiously.

System Testing

The period of time in the software lifecycle during which the implementation of each requirement is validated.

Tailoring

In the TPM process, this term refers to one of the two basic measurement activities, which comprise the system measurement process. The tailoring activity includes identification and prioritization of program issues, selection and specification of appropriate system measures, and integration of the measurement requirements to the developer's system process.
See **Application**.

Test

The process of exercising a product to identify differences between expected and actual behavior.

Test Artifacts

An item created during the system integration and test process that is preserved upon completion of the test process (e.g., test plans, requirements documentation, automated test scripts, and test documentation).

Test Case

A description of a test to be executed for or focused on a specific test aim.

Test Case Procedures

A sequence of steps, sub-steps, and other actions, performed serially, in parallel, or in some combination of consecution, that creates the desired test conditions that the test case is designed to evaluate.

Test Case (Setup) Suite

The steps required to configure the test environment for execution of a test case.

Testing Condition

System state or circumstance created by proceeding through some combination of steps, sub-steps, or actions in a test case.

Testing Environment

The infrastructure in which the test is performed, consisting of hardware, system software, test tools, and procedures.

Test Plan

In a test plan the general structure and the strategic choices with respect to the test to be executed are formulated. The test plan forms the scope of reference during execution of the test and also serves as an instrument to communicate with the customer of the test. The test plan is a description of the test project, including a description of the activities and planning, therefore it is *not* a description of the tests themselves.

Test Readiness Review (TRR)

Review conducted to determine whether a software test phase has been completed and to assure that the software is prepared for the next step in the formal integration and testing procedures. Software test procedures and results are evaluated, for compliance with the software testing requirements and system descriptions, for adequacy in accomplishing testing goals. Also, provides the forum for updating and revising operational and supporting documentation.

Test Resources

Aids that are used by a test tool for collecting, tracking and controlling information. This information is:

- Software requirements defined in the Software Requirements Document.
- Test requirements defined in the System Test Description.
- Automated test case scripts as defined in the System Test Description.
- SPRs as determined at each phase of the System Integration and Testing process.

This information is controlled by Configuration Management at the end of the SI&T process for use whenever further testing may be conducted, using a testing tool, during the remaining lifecycle of the software or system.

Test Tools

The software, hardware, systems, or other instruments that are used to measure and test an item.

Traceability

Degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor, successor, or master-subordinate relationship to one another (e.g., the degree to which the requirements and design of a given software component match).

Unit

The lowest element of a software hierarchy that contains one or more of the following characteristics:

(1) A unit comprising one or more logical functional entities.

(2) An element specified in the design of a computer software component that is separately testable.

(3) The lowest level to which software requirements can be traced.

(4) The design and coding of any unit can be accomplished by a single individual within the assigned schedule.

Unit Test

The process of ensuring that the unit executes as intended. This usually involves testing all statements and branch possibilities.

Version

One of a sequence of copies of a system, each incorporates new modifications.

Version Identifier

A unique identifier assigned to baseline software, documentation, and test environment.

Version Control

The process by which all changes to the software, documentation, and test environment are compiled and built into a new version of the system.

Version Control Report

A report that details all changes and enhancements made to current version of the software, documentation, and test environment.

White Box Testing

This type of testing is associated with structural testing in which the testing can be characterized as being tied to implementation details, such as control methods, database design, coding details, and logic paths. The process of how an individual input is treated to produce a given output is ascertained. Structural testing is sometimes referred to as "clear box testing" since white boxes are considered opaque and do not really permit visibility into the code.

Work Breakdown Structure (WBS)

A work breakdown structure for software defines the software-related elements associated with program work, work activities, and products. Many measures are aggregated and analyzed at various WBS levels.

BIBLIOGRAPHY

Black, Rex, *Managing The Testing Process,* Redman, WA: Microsoft Press, 1999

Koomen, Tim and Pol, Martin, *Test Process Improvement*, Essex, England, UK: Pearson Education Limited, 1999

Carnegie Mellon University, Software Engineering Institute, *The Capability Maturity Model: Guidelines for Improving the Software Process*, 1995.

Institute of Electrical and Electronics Engineers (IEEE), "Glossary of Software Engineering Terminology," IEEE-Std-610.12, 1990.

Institute of Electrical and Electronics Engineers (IEEE)/Electronic Industries Alliance (EIA), "Software Life Cycle Processes," IEEE/EIA Std-12207, 1998.

U.S. Department of Education, "SFA System Integration & Testing Approach, SFA Modernization," Undated.

Federal Systems Integration and Management Center (FEDSIM), "FEDSIM Writers Guide, Version 2," May 1994

Free On-Line Dictionary Of Computing web site at www.foldoc.org

This page intentionally left blank

APPENDIX A

SYSTEM TEST PLAN TEMPLATE

This page intentionally left blank

APPENDIX B

SYSTEM TEST DESCRIPTION TEMPLATE

This page intentionally left blank